Apuntes introductorios: Generando Arte Artificial con GAN’s

María Guadalupe Quijano Escalera

# RESUMEN Y OBJETIVO

Usar GAN´s (Redes Neuronales Generativas Adversarias) para producir imágenes pre etiquetadas que permitan entrenar modelos generando arte artificial a través de imágenes de cuadros de Monet. Haciendo uso del aprendizaje profundo y redes neuronales, se podrá explorar cómo es que el aprendizaje automático puede ser utilizado para comprender la interacción entre la tecnología y el arte.

# OBJETIVOS

1. Generar imágenes artificiales con resoluciones que pueden ser muy bajas a muy altas a través de un crecimiento progresivo de la capa y modificando la entrada de cada nivel por separado, lo que permite obtener mejoras en diferentes atributos de la imagen llegando a obtener detalles precisos y finos.

# VERIFICACIÓN

Usar un set de datos de imágenes que permite etiquetar aspectos de interés en los cuadros de Monet. Estos nos proporcionan la información que alimenta otra tarea que detectara los cuadros, esto servirá para el entrenamiento de otra red neuronal.

Redes Neuronales Artificiales

ÍNDICE

CONTENIDO

#### REDES NEURONALES

#### HISTORIA LAS REDES NEURONALES

#### PERCEPTRÓN

#### ARQUITECTURA DE LAS REDES

#### FUNCIÓN DE COSTO, ACTIVACIÓN Y PÉRDIDA

#### MODELOS DE REDES NEURONALES

#### EL MODELO DE MCCULLOCH & PITTS

#### RECURSOS

# **Introducción**

Estos apuntes están diseñados para personas que por primera vez se introducen en el tema de las redes neuronales artificiales. Se muestra de forma resumida la historia de las redes neuronales, el funcionamiento básico de una neurona, su partes, su arquitectura, elementos importantes que la constituyen así como de sus aplicaciones.

**Palabras clave**

Deep learning, aprendizaje, redes neuronales, inteligencia artificial y detección de objetos.

# 

# **Glosario**

1. **Algoritmo Genético.** Conjunto de procesos ordenados inspirados en la evolución, que se repiten por un número finito de iteraciones. El cual involucra tres segmentos de operadores: la votación, el cruzamiento y la mutación.

El primero selecciona a los cromosomas en la ciudad para la gestación. En el cruzamiento elige aleatoriamente una posición para trocar las secuencias de los padres y gestar una habitual. Y el tercero modifica aleatoriamente algunos bits en el cromosoma.

1. **Arquitectura**. Para una Red Neuronal Artificial es necesario conocer la topología, la conectividad y la función de transferencia. La topología se refiere al número de nodos y capas de la red.
2. **Inteligencia Artificial**. En general, se puede definir bajo dos grandes posturas.La primera se ocupa de la forma idealizada de cómo piensan o actúan los humanos. Y la segunda se ocupa de la forma racional de cómo piensan o actúan los humanos, donde un sistema es racional si este hace algo correcto, de acuerdo a lo que conoce.
3. **Neurona artificial.** Es unidad de procesamiento de la información, ante un vector de entradas proporciona una única salida. Sus elementos principales son:

* Conjunto de entradas, xj Pesos sinápticos, wi .
* Operación neuronal: w1·x1+ w2·x2 + ... + wn·xn = a .
* Función de transferencia: y = F (w1·x1+ w2·x2 + ... + wn·xn ) .
* Bias o polarización: entrada constante de magnitud 1, y peso b que se introduce en el sumador u operador.

1. **Nodos.** Unidades de una red neuronal.
2. **Red Neuronal Artificial.**Es un sistema computacional inspirado en las redes neuronales biológicas, el cual consiste en un conjunto de nodos, los cuales son los encargados de realizar un procesamiento simple.

# **Resúmen**

**¿QUÉ SON LAS REDES NEURONALES ARTIFICIALES?**

Las redes neuronales artificiales son sistemas paralelos computacionales (bio)inspirado en el comportamiento observado en las neuronas biológicas, estas tratan de resolver problemas, así como representar el conocimiento de un modo conexionista y adaptativo.Están compuestas por capas de unidades llamadas neuronas artificiales, donde se define un flujo de información que, a través de un entrenamiento en la red neuronal, se obtiene una capacidad para comprender y procesar información proporcionada por datos en la capa de entrada.

**ARQUITECTURA DE LAS REDES**

Dependiendo de cómo se definan las conexiones y operaciones en medio de las capas de una red neuronal artificial es que se procesa de forma distinta la información, lo cual posibilita que diversas arquitecturas se usan para labores concretas, como laborar con imágenes o audio, o laborar con escrito, etcétera.

**DATO HISTÓRICO**

Allan Turing, en 1936, fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación

El prototipo esquemático de una neurona artificial ha sido planteado por warren mcculloch y walter pitts en 1943, y luego en 1960 el modelo de “perceptrón” planteado por Frank Rosenblatt. Los modelos basados en neuronas artificiales fueron decayendo un poco, hasta finales de los 80’s y en los 90s, donde volvieron a triunfar mucho auge con nuevos modelos y operaciones definidas entre sus capas.

**APLICACIONES DE LAS REDES NEURONALES**

Las redes neuronales son utilizadas para resolver diferentes problemas y tienen diversas áreas de aplicación, por ejemplo;

* Procesamiento del lenguaje natural.
* Detección de objetos en imágenes y video.
* Generación de nuevos fármacos.
* Conducción en vehículos autónomos, y muchas cosas más..

# **Introducción**

Las redes neuronales, se inspiran en el comportamiento conocido hasta el momento del cerebro humano, primeramente al referido a las neuronas y sus conexiones, tratando de generar modelos artificiales que solucionen problemas difíciles de resolver a través de técnicas algorítmicas convencionales.

Su funcionamiento se apoya en una agrupación de neuronas conectadas entre sí y que trabajan en conjunto, sin que haya una orden concreta para cada una. Con la experiencia, las neuronas van creando y reforzando ciertas conexiones para "saber" algo que se queda paralizado.

Sin duda alguna, hoy en día son un campo muy importante dentro de la Inteligencia Artificial.. Las redes neuronales permiten explorar la estandarización de parámetros que mejor se ajusten a un problema, por lo cual las redime en un campo que promete avances a corto plazo, principalmente en reconocimiento de ilustración/ imagen y de sonido.

# **Historia**

**INTRODUCCIÓN BIOLÓGICA**

En 1888 Ramón y Cajal demuestran que el sistema nervioso está compuesto por una red de células individuales, las neuronas, ampliamente interconectadas entre sí. La información fluye desde las dendritas hacia el axón atravesando el soma.

Dato interesante:

•Existen entre 100 000 millones de neuronas en el cerebro y cerca de un trillón de células gliales, en el cuerpo celular o soma y llegan a medir entre 4 a 130 micras de longitud, y no hay dos iguales en cuanto a forma. Del cuerpo celular surge un denso árbol de ramificaciones (árbol dendrítico) formado por las dendritas.

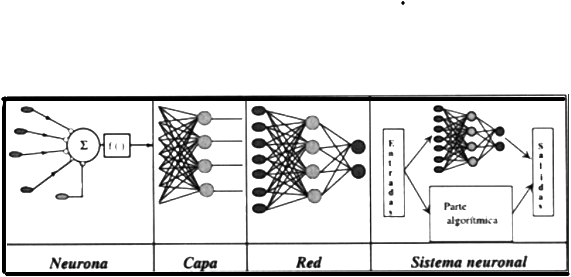


Figura 1. Modelo estándar del sistema global de proceso de una red neuronal.

El elemento básico de un sistema neuronal biológico es la *neurona*. Un sistema neuronal biológico está compuesto por millones de neuronas organizadas en capas.

En la emulación de dicho sistema neuronal biológico, por medio de un sistema neuronal artificial, se puede establecer una estructura jerárquica similar a la existente en el cerebro. El elemento esencial será la neurona artificial, la cual se organizará en capas.

Varias capas constituirán una red neuronal. Finalmente una red neuronal junto con los interfaces de entrada y salida constituirá el sistema global de proceso.

**RESEÑA HISTÓRICA DE LAS REDES NEURONALES**

La primera neurona artificial fue propuesta por Warren McCulloch y Walter Pitts en 1943, conocida como el modelo de McCulloch & Pitts (MCP). El modelo se conoce también como TLU (Threshold Logic Unit) o LTU (Linear Threshold Unit).

Dependiendo de la función de activación a la neurona puede ser lineal o no-lineal. Es decir, se genera un f = 1 en la salida si *Pn i=1 xiwi >* t,en caso contrario la neurona genera *f = 0*. Donde *w1, w2, . . . , wn* se conocen como pesos y *t* como el umbral (threshold).

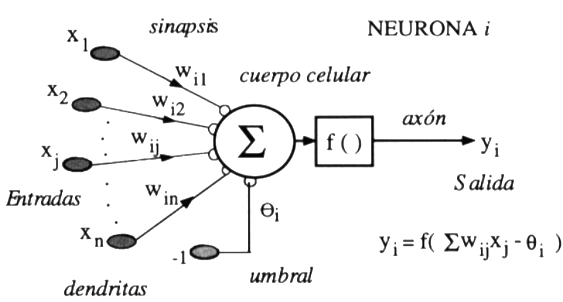


Figura 2: Modelo estándar de una neurona artificial.

Una regla de propagación *hi* es definida a partir del conjunto de entradas y los pesos sinápticos, dicha función de activación, la cual representa simultáneamente la salida de la neurona y su estado de activación. Si denotamos por yi dicha función de activación.

**PERCEPTRÓN**

Es la red neuronal más antigua. Es una forma de red neuronal de retroalimentación, en la que las conexiones entre los nodos no forman un bucle.

Acepta múltiples entradas, cada entrada se multiplica por un peso y los productos se suman. Los pesos simulan el papel de la *sinapsis* en las neuronas biológicas (para inhibir una señal). Luego, se agrega un valor de sesgo al resultado antes de pasarlo a una función de activación que va a simular el disparo de la neurona o no.

# **Elementos básicos que componen a una red neuronal.**

* **Función de entrada (input function).**
* **Función de activación (activation function).**
* **Función de salida (output function).**

# **Función de costo, activación y pérdida.**

**FUNCIÓN DE PÉRDIDA:**

Es un método que se ocupa para evaluar el modelo del algoritmo, así como su conjunto de datos.

Si presentan predicciones permanentemente erróneas, su funcionalidad de pérdida generará un número más grande. Por lo contrario, si estos son bastante buenos, generará un número menor. Mientras adaptará su algoritmo para intentar mejorar su modelo, su funcionalidad de pérdida le mencionara si está perfeccionando o no. La 'pérdida' nos ayuda a entender en qué medida el costo predicho difiere del costo real.

**FUNCIÓN DE COSTO:**

Es una representación que mide el rendimiento de un modelo de aprendizaje automático, siendo un proceso eficiente para datos dados. La función de costo es la que cuantifica el error entre los valores predichos y los esperados, se presenta en forma de un solo número real . Dependiendo del problema, la función de costo se puede generar de muchas formas distintas.

Introduciremos una función de coste con el fin de resolver y entrenar nuestro modelo.Se lleva a cabo cuando el error de pérdida es calculado para un solo ejemplo de entrenamiento. Si poseemos 'n' número de ejemplos, el promedio de la capacidad de pérdida de todo el grupo de entrenamiento es lo que se conoce como función de costo.

Su principal objetivo de la función de costo es:

* Maximizar:es el valor que se produce, se denomina recompensa . El propósito es colocar valores de parámetros de modelo para los que cifra el número más grande.
* Minimizar: es el valor que devuelto, se denomina costo, pérdida o error. El propósito es colocar valores de parámetros del modelo para que la función de costo devuelva el menor número posible.

**Diferencia entre la función de pérdida y de costo:**

La función de pérdida (o error) es para un solo ejemplo de entrenamiento. Por otro lado, tenemos a la función de costo, es la encargada de cubrir todo el grupo de entrenamiento (descenso del gradiente).

**FUNCIÓN DE ACTIVACIÓN:**

Una función de activación es un carácter muy notable de una red neuronal artificial, es la encargada de decidir si la neurona debe activarse o no. En las redes neuronales artificiales, la representación de la función de activación se encarga de definir la salida de dicho nodo dada una umbral o un conjunto de entradas.

**Tipos de funciones de activación:**

* Función lineal: la entrada de suma ponderada es proporcional a la salida, toma la forma; y = mx + c (m es la ecuación lineal que representa W yc se representa como b en redes neuronales, por lo que la ecuación se puede modificar como y = Wx + b ). Las entradas ( Xi ), multiplicadas por los pesos ( Wi ) de cada neurona, y crea una salida proporcional a la entrada.
* Función no lineal: se dividen principalmente en función de su rango o curvas.
* Función de paso binario: el gradiente (diferencial) de la función de paso binario es cero, que es el gran problema en el apoyo trasero para la actualización del peso.

**1. Sigmoide:** Los valores de salida se limitan entre 0 y 1, normalizando la salida de cada neurona. Permite predicciones claras.

**2. ReLU (unidad lineal rectificada):** Tiene una función derivada y permite la propagación hacia atrás. Es computacionalmente eficiente; permite que la red converja rápidamente

**3. Softmax:** Se usa solo para la capa de salida, para redes neuronales que necesitan clasificar las entradas en varias categorías. Es capaz de manejar múltiples clases solo una clase en otras funciones de activación. Normaliza las salidas para cada clase entre 0 y 1, y divide por su suma, dando la probabilidad de que el valor de entrada esté en una clase específica.

La regresión Softmax es una forma de regresión logística que normaliza un valor de entrada en un vector de valores. Los valores de salida están entre el rango [0,1].

# **Arquitectura de redes neuronales**

Se denomina arquitectura a la topología, estructura o patrón de conexión de una red neuronal. En una red neuronal artificial los nodos se conectan por medio de sinapsis, estando el comportamiento de la red determinado por la estructura de conexiones sinápticas. Estas conexiones sinápticas son direccionales, es decir, la información solamente puede propagarse en un único sentido (desde la neurona presináptica a la pos-sináptica). En general las neuronas se suelen agrupar en unidades estructurales que denominaremos capas. El conjunto de una o más capas constituye la red neuronal.

Se distinguen tres tipos de capas: de entrada, de salida y ocultas. Una capa de entrada, también denominada sensorial, está compuesta por neuronas que reciben datos o señales procedentes del entorno. Una capa de salida se compone de neuronas que proporcionan la respuesta de la red neuronal.

Una capa oculta no tiene una conexión directa con el entorno, es decir, no se conecta directamente ni a sensores ni a efectores. Este tipo de capa oculta proporciona grados de libertad a la red neuronal gracias a los cuales es capaz de representar determinadas características del entorno que trata de modelar.

**CAPAS**

* Capa de entrada: el número de neuronas en esta capa corresponde al número de entradas a la red neuronal. La capa consta de nodos pasivos, es decir, que no participan en el actual modificación de la señal, pero solo transmite la señal a la siguiente capa.
* Capas ocultas: estas capas se caracterizan por ser las intermedias entre la capa de entrada y la de salida, con número arbitrario de neuronas. Los nodos de esta capa toman parte en la modificación de la señal, por lo tanto, están activos.
* Capa de salida: la cantidad de neuronas en la capa de salida corresponde al número de los valores de salida de la red neuronal. Los nodos en esta capa son los activos.

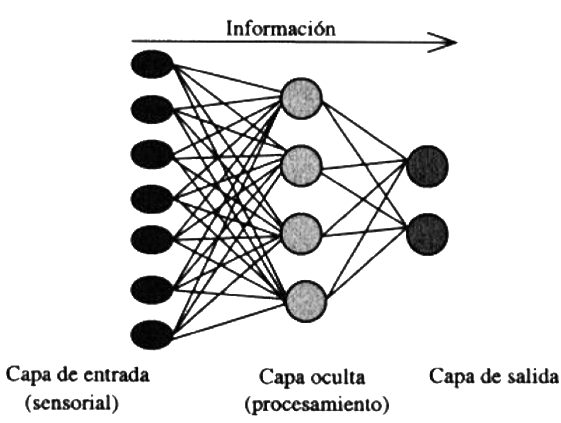


Figura 3: Arquitectura unidireccional con tres capas de neuronas: una capa de entrada, una capa.

# **Modelos de Redes**

**PERCEPTRÓN SIMPLE**

El perceptrón simple fue introducido por Rosenblatt (1962) y es un modelo unidireccional compuesto por dos capas de neuronas, una de entrada y otra de salida. La operación consiste en un perceptrón simple que consta de n neuronas de entrada y m neuronas de salida.

La importancia del perceptrón simple radica en el hecho de su carácter entrenable, ya que el algoritmo introducido por Rosenblatt (1962) permite que el Perceptrón simple determine automáticamente los pesos sinápticos que clasifican un conjunto de patrones etiquetados.

El algoritmo de aprendizaje del perceptrón simple pertenece al grupo de los algoritmos que se fundamentan en la corrección de errores. Los algoritmos de este tipo ajustan los pesos de manera proporcional a la diferencia existente entre la salida actual de la red neuronal y la salida deseada, con el objetivo de minimizar el error actual de la red.

**Backpropagation: Redes de propagación hacia atrás.**

En 1961, J. Kelly, Henry Arthur y E. Bryson introdujeron el concepto básico de backpropagation. En 1982, Hopfield presentó su idea de una red neuronal. Fue hasta 1993, cuando Wan fue la primera persona en ganar un concurso internacional de reconocimiento de patrones con la ayuda del método de backpropagation. En la minería de datos simplifica la estructura de la red al eliminar los enlaces ponderados que tienen un efecto mínimo en la red entrenada.

Es una técnica de descenso de gradientes. El descenso de gradiente tiene como objetivo encontrar un mínimo local de una función moviéndose iterativamente en la dirección opuesta del gradiente (es decir, la pendiente) de la función en el punto actual.

También conocida como *propagación hacia atrás de errores*; es el método que permite entrenar redes neuronales artificiales. Ayuda a calcular el descenso de gradientes de una función de pérdida con respecto a todos los pesos de la red, para ajustar los pesos de una red neuronal en función de la *tasa de error* obtenida en la época anterior (iteración). El ajuste adecuado de los pesos le permite reducir las tasas de error y hacer que el modelo sea confiable aumentando su generalización. El descenso de gradiente tiene como objetivo encontrar un mínimo local de una función moviéndose iterativamente en la dirección opuesta del gradiente (es decir, a la pendiente) de la función en el punto actual.

* Es un método flexible ya que no requiere conocimientos previos sobre la red.
* No tiene parámetros para sintonizar aparte de los números de entrada
* Es útil para las redes neuronales profundas que trabajan en proyectos propensos a errores, como el reconocimiento de imágenes o de voz.

**Tipos de redes de retropropagación**

1. **Backpropagation estática:**

Es un tipo de red de retropropagación que produce un mapeo de una entrada estática para una salida estática.

1. **Backpropagation recurrente:**

Es muy utilizado en la minería de datos, se adelanta hasta que se alcanza un valor fijo. Después de eso, el error se calcula y se propaga hacia atrás.

**Diferencia entre Backpropagation recurrente y estático**

El mapeo es rápido en la backpropagation estática mientras que no es estático en la Backpropagation recurrente.

**¿Cómo funciona el algoritmo de backpropagation?**

El algoritmo de retropropagación en la red neuronal *calcula el gradiente* de la función de pérdida para un solo peso mediante la regla de la cadena. Calcula de manera eficiente una capa a la vez. Calcula el gradiente, pero no define cómo se usa.

Generaliza el cálculo en la regla delta.

**PERCEPTRÓN MULTICAPA: Feedforward neural network**

También conocidas como "perceptrones multicapa" (MLP). Es una red neuronal artificial de retroalimentación en donde los nodos nunca forman un ciclo. Este tipo de red neuronal tiene una capa de entrada, capas ocultas y una capa de salida. Es el primer y más simple tipo de red neuronal artificial. Estas conexiones no son todas iguales, ya que cada conexión puede tener una fuerza o un peso diferente. Los pesos de dichas conexiones se encargan de codificar el conocimiento de una red.

**¿Cómo funciona una red neuronal Feedforward?**

Este modelo se trata de un perceptrón de una sola capa, que cuenta con una serie de entradas que ingresan a la capa y se multiplican por los pesos. Luego, cada valor se sumará para obtener la suma de los valores de entrada ponderados. Dicho esto, si la suma de los valores está por encima de un umbral específico (establecido en cero), el valor producido tendrá que ser 1, mientras que si la suma cae por debajo del umbral, el valor de salida es -1.

**Aplicaciones de las redes neuronales Feedforward**

Su arquitectura simplificada se puede utilizar como una ventaja en determinadas aplicaciones de aprendizaje automático. Usando una propiedad conocida como la regla delta, la red neuronal puede comparar las salidas de sus nodos con los valores previstos, permitiendo así que la red ajusta sus pesos a través del entrenamiento para producir valores de salida más precisos. Este proceso de formación y aprendizaje produce una forma de descenso en gradiente.

# 

Figura 4: Arquitectura de las redes neuronales artificiales según su dimensión.

**Asociaciones entre la información de entrada y salida.**

La información a información se registra de forma distribuida en los pesos asociados a las conexiones entre neuronas, por lo que, los datos que se guardan o almacenan en dicha memoria tienen como resultado el aprendizaje de la red y que podrán ser leídos a la salida como respuesta a cierta información de entrada, permitiendo que la red funcione con una memoria asociativa ya que se aplica un estímulo que será el dato de entrada la red y que responde con una salida asociada a dicha información de entrada.

# **Optimización**

En esencia, el objetivo del entrenamiento de redes neuronales es minimizar la función de coste encontrando los pesos adecuados para las aristas de la red . El descubrimiento de estos pesos se lleva a cabo mediante un algoritmo numérico llamado backpropagation.

El optimizador, es el encargado de generar pesos cada vez mejores: su importancia es crucial. Su funcionamiento esencial se basa en calcular el gradiente de la función de coste (derivada parcial) por cada peso (parámetro/dimensión) de la red. Como queremos minimizar el error, modificaremos cada peso en la dirección (negativa) del gradiente.

El conjunto de métodos iterativos de reducción de la función de error (búsqueda de un mínimo local), son conocidos cómo los método de optimización basados en el gradiente descendente.



Figura 5: Cálculo del peso en el paso t+1

**Gradient Descent**

El método de descenso por gradiente es uno de los algoritmos de optimización más populares en aprendizaje automático, particularmente por su uso extensivo en el campo de las redes neuronales. Gradient descent es un método general de minimización para cualquier función. A la versión original se le considera lenta pero versátil, sobretodo para casos de que la funciones multi-dimensionales.

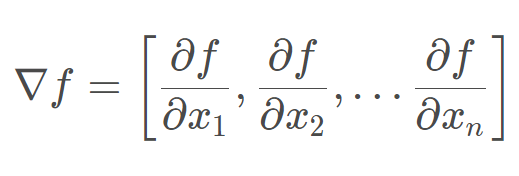
El gradiente es la generalización vectorial de la derivada, es un vector de tantas dimensiones como la función y cada dimensión contiene la derivada parcial en dicha dimensión:

Figura 6: Forma matemática del gradiente.

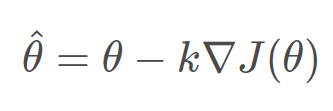
En aprendizaje automático usamos gradient descent para encontrar los parámetros de nuestro modelo que mejor definen nuestro conjunto de entrenamiento. La forma de lograr eso es minimizar la función de costo ***J*** de la siguiente forma:

Figura 7: Definición del descenso de gradiente.

Donde ***θ*** acento circunflejo es la nueva posición de parametros, ***θ*** los parametros anteriores y ***k*** la tasa de aprendizaje o learning rate

**Stochastic Gradient Descent (SGD)**

El problema con el descenso de gradiente es que la convergencia a un mínimo local lleva mucho tiempo y no se garantiza la determinación de un mínimo global. En SGD, el usuario inicializa los pesos y el proceso actualiza el vector de peso usando un punto de datos. El descenso de gradiente lo actualiza continuamente de forma incremental cuando se completa un cálculo de error para mejorar la convergencia.

El método busca determinar el descenso más pronunciado y reduce el número de iteraciones y el tiempo necesario para buscar grandes cantidades de puntos de datos. En los últimos años, el tamaño de los datos ha aumentado enormemente, de modo que las capacidades de procesamiento actuales no son suficientes. El descenso de gradiente estocástico se utiliza en redes neuronales y reduce el tiempo de cálculo de la máquina al tiempo que aumenta la complejidad y el rendimiento para problemas a gran escala.

**ADAM**

Adam es un método de tasa de aprendizaje adaptativo, lo que significa que calcula las tasas de aprendizaje individuales para diferentes parámetros. Su nombre se deriva de la estimación del momento adaptativo, y la razón por la que se llama así es porque Adam usa estimaciones del primer y segundo momento del gradiente para adaptar la tasa de aprendizaje para cada peso de la red neuronal. La forma matematica es algo elavorada y la explicación sale de los alcances de este texto, sin embargo es recomendable poder estudiarlo y entenderlo de la mejor forma posible.

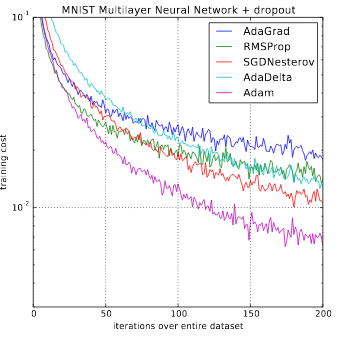


Figura 7: Comparación del funcionamiento de diferentes optimizadores

# **Aplicaciones de las redes neuronales**

Las redes neuronales son utilizadas para resolver diferentes problemas, ya que pueden ser aplicadas en diversas áreas, por ejemplo;

* Empresa:Explotación de bases de datos, reclutamiento de empleados, reconocimiento de caracteres escritos, optimización de horario, etc.
* Finanzas:Identificación de falsificaciones y firmas, valoración del riesgo de los créditos, etc.
* Medio ambiente: Analizar tendencias, tipologías y patrones,etc.
* Medicina: Monitorización en cirugías, diagnóstico y tratamiento a partir de síntomas o de datos analíticos, predicción de reacciones adversas en los medicamentos, etc.

# 

# **RECURSOS**

Bibliográficas complementaria.

**LIBRO:**

Fernando Berzal: Redes Neuronales & Deep Learning. Edición independiente, 2018

ISBN 1-7312-6538-7 (b&n)

ISBN 1-7313-1433-7 (color)

https://deep-learning.ikor.org

**LECTURAS:**

* Convolutional Neural Networks for Visual Recognition. Andrej Karpathy (Stanford University CS231n) http://cs231n.github.io/
* Deep Learning: Methods and Applications. Li Deng & Dong Yu (Microsoft Research)<http://research.microsoft.com/apps/pubs/default.aspx?id=209355>
* https://atcold.github.io/pytorch-Deep-Learning/es/

**SIMULADORES:**

* <http://demos.algorithmia.com/classify-places/>
* [http://playground.tensorflow.or](http://playground.tensorflow.org)g

[Black and White Vertical Presentation (canva.com)](https://www.canva.com/design/DAEjXwAlbRY/-Nfu_nN7Jczy1rluo3WKOA/view?utm_content=DAEjXwAlbRY&utm_campaign=designshare&utm_medium=link&utm_source=homepage_design_menu&fbclid=IwAR1V9L2ODrBf-34IwpyFk9NQEitA1TyS27dmu6ksTmeTg88fDiZZ38pL6FM#14)